



Pertemuan 11 : PHP OOP

Pemrograman Web

Agung Nugroho, M.Kom

Teknik Informatika – S1

Fakultas Teknik

Universitas Pelita Bangsa



Agung Nugroho, S.Kom, M.Kom

- 1994 | SDN Pulau Panggung, OKU Sumsel
- 1997 | MTs Lab Fak Tarbiah IAIN SUKA, Yogyakarta
- 2000 | SMK PIRI 1, Yogyakarta
- 2004 | Ilmu Komputer, Universitas Ahmad Dahlan, Yogyakarta
- 2016 | Magister Komputer, STMIK Eresha, Jakarta

- 2012 - Present | Freelance Web Developer
- 2011 - 2012 | Web Developer at BP Indonesia
- 2010 - 2011 | OSS Core Engineer at PT Ericsson Indonesia
- 2008 - 2009 | Radio Database Planner at PT. NextWave subcon NSN
- 2005 - 2008 | Software Developer at PT Gamatechno Indonesia
- 2004 - 2005 | Web Programmer at PT Reftindo Sarana



- www.linkedin.com/in/kangmasagung
- www.fb.me/agung.n
- www.koding.web.id

Object Oriented Programing in PHP

Pertemuan 11



Apa itu OOP?

- OOP (*Object Oriented Programming*) adalah suatu metode pemrograman yang berorientasi kepada objek.
- Tujuan dari OOP diciptakan adalah untuk mempermudah pengembangan program dengan cara mengikuti model yang telah ada di kehidupan sehari-hari.



Object

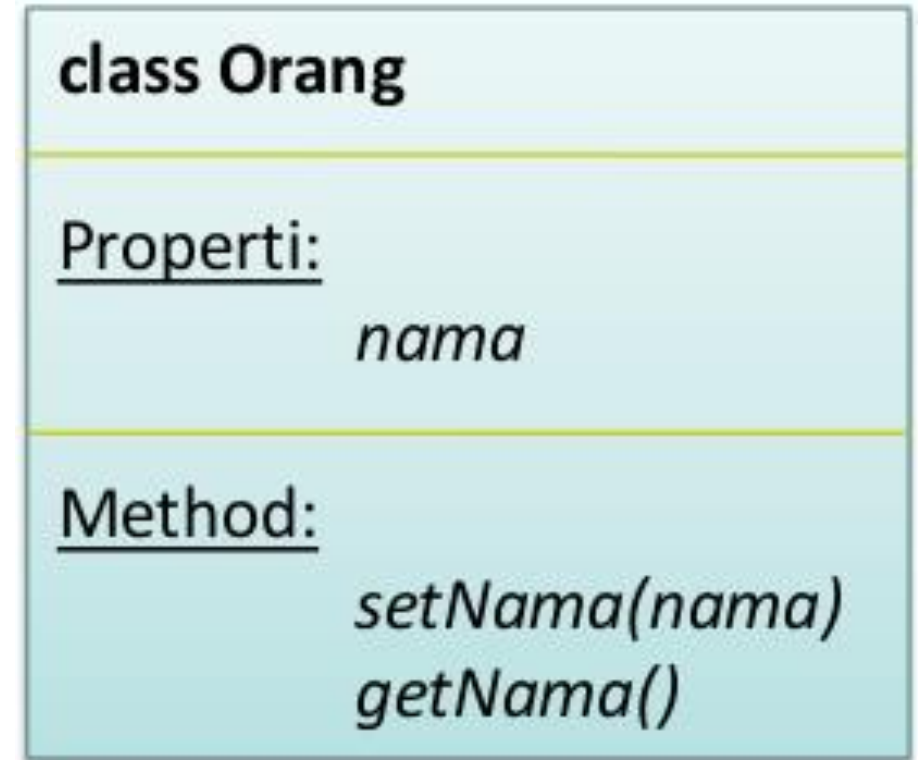
Perbedaan utama OOP dengan pemrograman terstruktur (fungsional), data dan kode program tergabung menjadi satu entitas yang disebut Objek.

- Tiap objek dapat berinteraksi satu dengan lainnya.
- Tiap objek biasanya mewakili satu persoalan, yang memiliki properti/atribut dan method.
 - Properti → Data
 - Method → Fungsi



Class

- Class adalah script yang digunakan sebagai cetakan untuk membuat objek.
- Class mendefinisikan properti/atribut yang dimiliki objek serta method yang dapat dilakukan oleh objek.



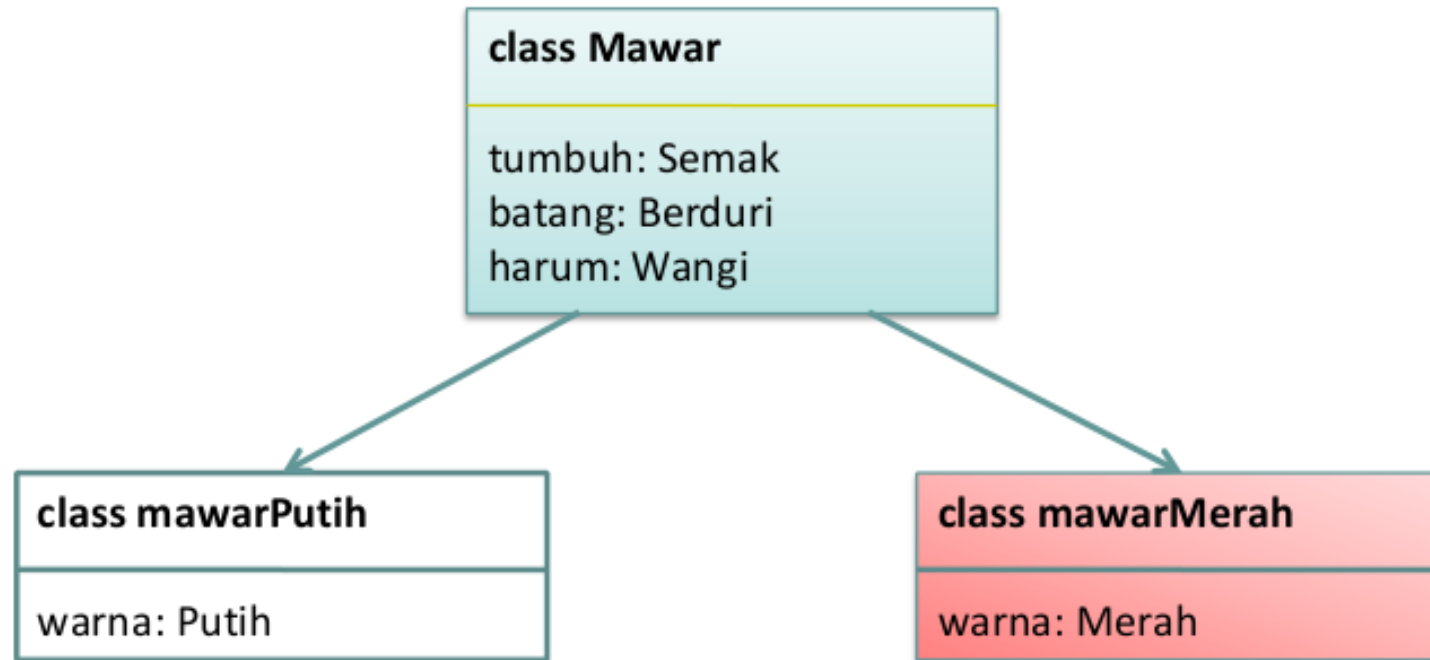


Inheritance

- Setiap objek harus memiliki properti dan method yang dibutuhkan. Tidak kurang, tidak lebih.
- ***Inheritance*** dapat memberikan properti dan method ke class-class lainnya.
- Misal ada 2 objek mawar, **mawarPutih** dan **mawarMerah**.
 - Keduanya memiliki beberapa informasi yang sama, tumbuh di semak-semak, berduri, harum.
 - Dengan *inheritance*, memungkinkan menghapus duplikasi objek.



Inheritance



- class Mawar disebut Parent class.
- class mawarPutih dan mawarMerah disebut Child class.
- Child class mewarisi properti dan method parent class



Modifier

3 keyword modifier yang dipakai banyak bahasa pemrograman

- **Public**

Properti dan Method dapat diakses oleh kelas itu sendiri dan oleh kelas lain melalui objeknya.

- **Protected**

Properti dan Method hanya dapat diakses oleh kelas itu sendiri dan kelas turunannya.

- **Private**

Propertii dan Method hanya dapat diakses oleh kelas itu sendiri.



Daur Hidup Objek

- Dibangkitkan
 - *Construct*
- Digunakan
- Dimusnahkan
 - *Destruct*

Deklarasi Class

- Sintaks

```
class NamaKelas {  
    // deklarasi properti  
  
    // deklarasi method  
}
```

- Contoh

```
class Orang {  
    // deklarasi properti  
    protected $nama;  
  
    // deklarasi method  
    public function setName($nama) {  
        $this->nama = $nama;  
    }  
  
    public function getName() {  
        return $this->nama;  
    }  
}
```



\$this

- Variabel yang digunakan untuk mengacu ke properti atau method di dalam kelas itu sendiri.
- Tidak dapat digunakan di luar kelas.
- Sintaks
 - `$this->namaProperti`
 - `$this->namaMethod([parameter])`



Penggunaan Objek

- Instansiasi objek dengan keyword new, diikuti nama Kelas.
- Contoh:

```
$budi = new Orang();  
$budi->setNama('Budi');  
echo "Nama saya: " . $budi->getNama();|
```




Inheritance

- Contoh

```
class Mahasiswa extends Orang {  
    private $nim;  
  
    public function setNim($nim) {  
        $this->nim = $nim;  
    }  
  
    public function getNim() {  
        return $this->nim;  
    }  
}
```

```
$mhs = new Mahasiswa();  
$mhs->setNama('Budi');  
$mhs->setNim('030429001');  
echo "Nama : " . $mhs->getNama() . "</br />";  
echo "NIM : " . $mhs->getNim();
```



Overriding

- Contoh

```
class Mahasiswa extends Orang {  
    private $nim;  
  
    public function setNim($nim) {  
        $this->nim = $nim;  
    }  
  
    public function getNama() {  
        return "<b>" . $this->nama . "</b>";  
    }  
  
    public function getNim() {  
        return $this->nim;  
    }  
}
```



Konstanta Kelas

- Deklarasi konstanta dengan keyword `const`
- Pemanggilan konstanta melalui kelasnya, bukan objeknya.
- Contoh

```
class Warna {  
    const MERAH = "Merah";  
    const HIJAU = "Hijau";  
    const BIRU = "Biru";  
  
    public function cetakBiru() {  
        echo self::BIRU;  
    }  
}
```

```
echo Warna::MERAH;  
$objWarna = new Warna();  
$objWarna->cetakBiru();
```



parent:: dan self::

- **self::**

- Mengacu ke kelas itu sendiri.
- Biasa digunakan untuk mengakses konstanta.

- **parent::**

- Mengacu ke kelas induk.
- Sering dipakai untuk memanggil konstruktor dari kelas induk.
- Dapat juga untuk mengakses konstanta.
- Gunakan parent:: untuk mengantisipasi bila terjadi perubahan hirarki kelas.



parent:: dan self::

```
class Induk {  
    const NAMA = "Induk";  
  
    public function __construct() {  
        echo "Konstruktor di kelas <b>" . self::NAMA "</b>";  
    }  
}  
  
class Anak extends Induk {  
    const NAMA = "Anak";  
  
    public function __construct() {  
        parent::__construct();  
        echo "Konstruktor di kelas <b>" . self::NAMA "</b>";  
    }  
}  
  
$obj = new Anak();
```




Constructor

- Constructor adalah method khusus yang tereksekusi otomatis ketika objek dari kelas tersebut dibuat.
- Constructor harus diberi nama : **`__construct()`**
- Tidak wajib membuat constructor



Constructor

```
class Orang {  
    // deklarasi properti  
    protected $nama;  
  
    // deklarasi method  
    public function __construct($nama = null) {  
        $this->nama = $nama;  
    }  
  
    public function setName($nama) {  
        $this->nama = $nama;  
    }  
  
    public function getName() {  
        return $this->nama;  
    }  
}
```

```
$budi = new Orang('Budi');  
echo "Nama saya: " . $budi->getName();
```



Destructor

- Method khusus yang dipanggil otomatis oleh PHP saat objek dimusnahkan.
- Destructor harus diberi nama : **`__destruct()`**
- Tidak wajib membuat method destructor.
- Buat method destructor jika ada script khusus yang ingin dieksekusi sebelum objek dimusnahkan.
- `unset($objek)`, untuk memusnahkan objek.
 - Tidak wajib digunakan, karena PHP otomatis memusnahkan objek yang tidak terpakai.



Destructor

```
class Orang {  
    public function __construct() {  
        echo "Object yang dibuat";  
    }  
  
    public function __destruct() {  
        echo "Object yang dimusnahkan";  
    }  
}
```



Question?



CodeLabs | Experiment



Persiapan

Tools:

- **Text Editor:** Sublime Text, VS Code, Atom, Dll
- **Web Browser:** Google Chrome, Mozilla Firefox, Dll
- **Web Server:** Apache, IIS, NGINX, Dll



Terimakasih

Agung Nugroho

agung@pelitabangsa.ac.id

www.koding.web.id